UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/001,478	11/01/2001	Craig Nemecek	CYPR-CD01213M	6435

45545 7590 06/04/2008

CYPRESS C/O MURABITO, HAO & BARNES LLP
TWO NORTH MARKET STREET
THIRD FLOOR
SAN JOSE, CA 95113

EXAMINER	
PROCTOR, JASON SCOTT	
ART UNIT	**PAPER NUMBER**
2123	
MAIL DATE	**DELIVERY MODE**
06/04/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/001,478	NEMECEK ET AL.	
	Examiner	Art Unit	
	JASON PROCTOR	2123	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *28 April 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,2,4-10,12 and 14-23* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,2,4-10,12 and 14-23* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claims 1, 2, 4-10, 12, and 14-23 were rejected in the Office Action entered on 30 January 2008.

Applicants filed a request for reconsideration on 28 April 2008. Claims 1, 2, 4-10, 12, and 14-23 are pending in this application.

Priority

1. This Application contains a claim for the benefit of priority to U.S. Provisional Application No. 60/243,708 filed 26 October 2000. The provisional application has been reviewed and priority is denied, because the provisional application does not appear to enable the claimed invention as required under 35 U.S.C. Section 112, first paragraph. See 35 U.S.C. § 119(e)(1).

For example, the provisional application contains a set of 'powerpoint-style' drawings and datasheets describing desired features for a microcontroller or a 'system-on-chip,' but this material does not appear to contain either the text description or the drawings found in the Application. In particular, no part of the provisional application appears to disclose the method steps shown in the Application at Fig. 7.

Response to Arguments – 35 USC § 103

2. Applicants' have stated for the record:

> Dictionary meanings of "dummy code" clearly show that the ordinary and customary meaning of "dummy code" is different from "booting code" recited in Claim 1. Unlike "booting code" which is used to boot the microcontroller, "dummy code" does not perform any functions to boot the virtual microcontroller.

> "Dummy code" is used in one embodiment to synchronize the virtual controller with the microcontroller using a timing loop. (Applicants' remarks, 31 October 2007, page 11)

Applicants' argue in the current remarks primarily that:

> [A]ccording to the citations, Coker fails to teach or suggest the claimed features that during the booting process, the microcontroller executes the set of boot code, whereas the virtual microcontroller executes the set of timing code. Unlike [Coker] where the same software is used for both the target-ECS and shadow system during the start up, Claim 1 recites the features that the microcontroller (which corresponds to the target-ECS in [Coker]) and the virtual microcontroller (which corresponds to the shadow system) do not run the same software.
>
> [...] Moreover, the rejection states that Teramoto teaches the claimed features that the dummy code is a set of timing code to take the same number of clock cycles as the microcontroller. In support of the rejection, the Examiner cites passages in Teramoto that "[t]he present invention relates to an instruction execution control method and information processing information apparatus for monitoring information about the completion of synchronization between processor, and selectively causing a specific instruction of the subsequent group of instructions to wait until the completion of synchronization is indicated when the processors are operated in synchronism with each other for synchronous execution of their respective processes in a computer system including a plurality of processors" [cit. omitted] and that "[w]hen the instruction, which was read out, is a wait instruction, the instruction analyzer 102 issues a wait instruction to the wait instruction controller 100 through the interface signal 105" where "[t]he wait instruction controller 100 executes the wait instruction which controls the instruction execution sequence according to the present invention," [cit. omitted].
>
> However, Applicants respectfully submit that no where in Teramoto teaches or suggests the features that during the booting process, the virtual microcontroller runs dummy code where the dummy code is a set of timing code to take the same number of clock cycles as the microcontroller, as claimed. (All emphasis added by Applicants)

The Examiner respectfully traverses this argument as follows.

In the 31 October 2007 remarks, Applicants deliberately stated for the record that "'Dummy code' is used in one embodiment to synchronize the virtual controller with the microcontroller using a timing loop." A "timing loop" is well known in the computer programming art as a set of instructions that loop, pause, or wait for an appropriate timing signal before resuming normal instruction execution. The Examiner interpreted the claimed "dummy code" according to the claim language (which does not exclude a timing loop) and specifically according to Applicants' statements for the record. The Teramoto reference discloses a timing loop mechanism as summarized above in Applicants' remarks.

Therefore, contrary to Applicants' conclusion, the Teramoto reference discloses the claimed dummy code as shown by Applicants' remarks that dummy code uses a "timing loop" to synchronize the two processors. Teramoto clearly discloses a method of synchronizing two processors [*"The present invention relates to an instruction execution control method and information processing apparatus for monitoring information about the completion of synchronization between processors, and selectively causing a specific instruction of the subsequent group of instructions to wait until the completion of synchronization is indicated when the processors are operated in synchronism with each other for synchronous execution of their respective processes in a computer system including a plurality of processors."* (Teramoto, column 1, lines 7-16); *"When the instruction, which was read out, is a Wait instruction, the instruction analyzer 102 issues a Wait instruction to the Wait instruction controller 100 through the interface signal 105. The Wait instruction controller 100 executes the Wait instruction which controls the instruction execution sequence according to the present invention."* (Teramoto, column 5, lines 50-60)].

In order to overcome this interpretation "dummy code," Applicants must explicitly amend the claim language to exclude the 31 October 2007 remarks or otherwise make a clear and deliberate disavowal of claimed subject matter. Applicants may not first argue that one embodiment employs a "timing loop" to achieve the claimed "dummy code" and then distinguish the claimed invention over the prior art which teaches the same timing loop.

Regarding the Coker reference, the Examiner acknowledges that Coker describes the two very different computer architectures as operating "the same software," however, a person of ordinary skill in the art would immediately recognize that the two different architectures described by Coker ("target-ECS" and "shadow system") may execute the same resulting

software but only by executing instructions specially adapted for each architecture. Coker explicitly describes how the two different processors execute "the same software" in very different architectures [*"Thus, rather than receiving input data from an external source and reading the data into complex I/O registers, the shadow system uses the data value and relative time of input events from the target-ECS and writes the value directly to its RAM using its internally generated location."* (Coker, column 2, line 63 – column 3, line 1)]. That is, Coker discloses to a person of ordinary skill in the art that the target-ECS has instructions for receiving data from an external source and reading the data into complex I/O registers, whereas the shadow system has instructions for receiving data from the target-ECS and writes the values directly into RAM. These different instructions are necessary in order for these two processors operating in different architectures to execute "the same software". Thus, in Coker, the target-ECS and the shadow system execute different instructions.

In conclusion, Coker describes that the target-ECS and shadow system execute "the same software" but a person of ordinary skill in the art would recognize from Coker's disclosure that these two processors must execute similar, but different sets of instructions in order to implement Coker's invention. Teramoto discloses "dummy code" using a timing loop mechanism as claimed, and further supported by Applicants' remarks that "dummy code" may employ a timing loop in order to achieve synchronization.

Applicants' arguments have been fully considered but have been found unpersuasive.

Applicants further argue that:

> [From Coker's] teaching that "[a] shadow system of this invention executes the same software as the target-ECS from system start-up or reset," [cit. omitted] one skilled in the art can infer that both the target-ECS and the shadow system execute the same boot code during the start up. Therefore, [Coker] at least does not

> teach the claimed features that the virtual microcontroller, which executes the timing code, cannot access the boot code during start up, as claimed.

The Examiner respectfully traverses this argument as follows.

Executing the same code as another processor is a completely different concept from accessing the instructions stored on the second processor. Applicants' have deliberately claimed that the boot code in the microcontroller is "inaccessible" to the virtual microcontroller. Coker has disclosed no means whatsoever that suggests that the shadow system can access the instructions on the target-ECS. Therefore, the instructions on the target-ECS are "inaccessible" to the shadow system. Whether or not these two processors execute the exact same instructions (they do not) is a separate issue.

Applicants' arguments have been fully considered but have been found unpersuasive.

Applicants refer to the arguments addressed above in favor of claims 10 and 12. These arguments have been fully considered but found unpersuasive.

Regarding claim 22, Applicants argue primarily that:

> Although Matyas teaches the passing of serial number and password as a part of the code being passed, the reference does not teach that any algorithm is contained in the code being passed. The third parameter P3 being passed is a computer number interpreted by the DES algorithm, See Column 13, Lines 13-15. Matyas does not teach that the DES algorithm, or any algorithm, is passed along with the serial number and password as part of the code.

The Examiner respectfully traverses this argument as follows.

The claim does not require "passing" any code containing algorithms. The claim requires "executing a set of boot code to carry out the initialization" (Claim 1) and "wherein the set of boot code comprises proprietary information, wherein the proprietary information comprises serial numbers, passwords, and algorithms." Matyas clearly teaches algorithms as shown in the

rejection. Additionally, Coker teaches computer software comprising algorithms [*"A shadow system of this invention executes the same software as the target-ECS from system start-up or reset."* (Coker column 2, lines 56-58)]; Rosenberg teaches computer software comprising algorithms [*"Debuggers are quite complex pieces of software. Their inner workings require a suite of sophisticated algorithms and data structures to accomplish their tasks."* (Rosenberg, page 2)]; Teramoto teaches computer software comprising algorithms (Teramoto, FIG. 10). The claimed boot code comprising *inter alia* algorithms would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention in view of the references.

Applicants' arguments have been fully considered but have been found unpersuasive.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. § 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. § 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. § 103(a), the examiner presumes that the subject matter of the various

claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. § 103(c) and potential 35 U.S.C. § 102(e), (f) or (g) prior art under 35 U.S.C. § 103(a).

3. Claims 1-2, 4-10, 12, 14-20, and 23 are rejected under 35 U.S.C. § 103(a) as being unpatentable over US Patent No. 5,371,878 to Coker in view of "How Debuggers Work" by Jonathan B. Rosenberg (Rosenberg), further in view of US Patent No. 5,968,135 to Teramoto et al.

Regarding claim 1, Coker teaches a method comprising:

In a microcontroller, executing a set of boot code to carry out initialization ("Embedded Computer System or ECS", see column 1, lines 20-23) [*"[T]his invention uses hardware and software which can 'shadow' the execution of a target-ECS in real time operation"* (column 2, lines 34-40); *"A shadow system of this invention executes the same software as the target-ECS from system start-up or reset."* (column 2, lines 56-58); Applicants' remarks state that: "Booting is a process that starts a system (e.g., operating system) and substantially initialize the system when a user turns on a computing system or a system with a processor." (Applicants' remarks, 6 February 2007, page 15)];

In a virtual microcontroller ("shadow system"), executing a set of timing code to enable the lock-step synchronization [*"A shadow system of this invention executes the same software as the target-ECS from system start-up or reset."* (column 2, lines 56-58); *"The shadow system and the target-ECS function exactly the same except that the shadow system receives data slightly*

delayed because of a data buffer between the target-ECS and the shadow system." (column 3, lines 13-16); *"[I]n terms of relative time, the execution state of the shadow system 28 at the time when any given instruction is executed will directly correspond to the execution state of the target-ECS when the same instruction was executed.."* (column 8, lines 44-49)]

and wherein the set of timing code is different from said set of boot code, [*"Thus, rather than receiving input data from an external source and reading the data into complex I/O registers, the shadow system uses the data value and relative time of input events from the target-ECS and writes the value directly to its RAM using its internally generated location."* (column 2, line 63 – column 3, line 1)];

and wherein the set of boot code is stored within the microcontroller and the set of boot code is inaccessible to the virtual microcontroller [Interface means 19 connecting the target-ECS and shadow system is used by Coker to transmit I/O data and does not appear to contain any suggestion that instruction code or boot code is transmitted via interface means. See (column 4, lines 9-18)].

Although Coker teaches "a computer system and method for debugging, verifying and developing an embedded computer system" (column 1, lines 9-11), Coker does not expressly teach old and well-known tools and techniques of debugging. As a result, Coker teaches a method of debugging, but does not expressly teach simultaneously halting both the microcontroller and the virtual microcontroller.

Rosenberg expressly teaches halting multiple processors operating in lockstep [*"A significant issue for debuggers on multiprocessor systems is how or whether other processors stop when a fault occurs in one. On SIMD architectures, by definition, all processors operate in lock step and so this is guaranteed."* (page 45, first full paragraph)].

Rosenberg and Coker are analogous art because both are drawn to debugging.

It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine the teachings of Rosenberg and Coker because Rosenberg provides teachings specifically directed to debugging tools, which are suggested by Coker, but "are very difficult tools to build robustly because they depend heavily on relatively weak portions of operating systems and because they tend to stress the underlying operating system's capabilities." (Rosenberg, page 1, first paragraph). That is, Rosenberg provides the teachings that provide helpful guidance in making and using the "critical tools for the development of software" (Rosenberg, page 1, first paragraph) that are suggested by Coker (title, abstract). The combination could be achieved as expressly taught by Rosenberg, by halting the target-ECS and the shadow system "simultaneously" because Rosenberg explicitly teaches this behavior in architectures, like the one claimed, that operate in lock step. (*Id.*)

Coker in view of Rosenberg does not expressly teach that the timing code is a dummy code timed to take the same number of clock cycles as the microcontroller uses to execute the set of boot code. Applicants have defined "dummy code" as exemplified in the remarks submitted on 31 October 2007.

Teramoto teaches timing code that is a dummy code timed to take the same number of clock cycles as a second processor executing code [*"The present invention relates to an instruction execution control method and information processing apparatus for monitoring information about the completion of synchronization between processors, and selectively causing a specific instruction of the subsequent group of instructions to wait until the completion of synchronization is indicated when the processors are operated in synchronism with each other for synchronous execution of their respective processes in a computer system including a*

plurality of processors." (column 1, lines 7-16); *"When the instruction, which was read out, is a Wait instruction, the instruction analyzer 102 issues a Wait instruction to the Wait instruction controller 100 through the interface signal 105. The Wait instruction controller 100 executes the Wait instruction which controls the instruction execution sequence according to the present invention."* (column 5, lines 50-60)].

Teramoto and Rosenberg in view of Coker are analogous art because both are drawn to operating processor synchronously.

It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine the teachings of Teramoto with Rosenberg in view of Coker as expressly motivated by Teramoto to maintain high execution speed [*"An object of the present invention is to provide an instruction execution control method and an information processing apparatus for enabling synchronized operations to be performed at high speed among a plurality of processors sharing a main storage."* (column 2, lines 40-44)].

Therefore it would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine the teachings of Teramoto with Rosenberg in view of Coker to achieve the invention specified in claim 1.

Regarding claim 2, Coker teaches copying register contents from the microcontroller to corresponding registers in the virtual microcontroller after completion of the simultaneous halting [*"An execution state, including the contents of RAM and I/O registers in the target-ECS and RAM and I/O state memory in the shadow system, can further be defined as including an input state vector, output state vector, and internal state vector. Input and output state vectors are defined as the contents of input and output registers respectively in the target-ECS and as the*

contents of the I/O memory state in the shadow system." (column 8, lines 50-57); copied from the target-ECS to the shadow system, *"the shadow system receives its input data from the input registers of the target-ECS and stores the input data in its RAM."* (column 2, lines 61-63)].

Also, Rosenberg teaches that *"debuggers provide disassembly views and direct access to hardware registers."* (page 6, first full paragraph).

Regarding claim 4, Rosenberg teaches that after executing code, a processor branches to an instruction line [*"A breakpoint is a special code placed in the executing code stream that, when executed, causes a special trap to occur that the processor and the operating system report to the debugger."* (page 5, second full paragraph); More generally at pages 56-57, "Generic OS-Debugger Interaction Model", etc.].

Regarding claim 5, Rosenberg teaches that prior to executing code, a break is set at an assembly instruction line [*"A breakpoint is a special code placed in the executing code stream that, when executed, causes a special trap to occur that the processor and the operating system report to the debugger."* (page 5, second full paragraph)].

Regarding claim 6, Coker teaches that the boot code comprises protected initialization code that is not accessible to the In-Circuit Emulation system [Coker teaches no means by which the in-circuit emulation system may access the code on the target-ECS. Here Applicants' have presented a negative limitation, which has been interpreted and treated according to MPEP 2173.05(i).].

Claim 7 recites a combination of limitations recited by claims 4 and 5, which are taught by Coker in view of Rosenberg, further in view of Teramoto as shown above.

Claim 8 recites a combination of limitations recited by claims 2, 4, and 5, which are taught by Coker in view of Rosenberg, further in view of Teramoto as shown above.

Regarding claim 9, Rosenberg teaches removing a break at an assembly line [*"Inactive breakpoints are place holders that the user can turn active but currently will not cause execution to stop if reached."* (page 27)].

Claim 10 recites a combination of limitations recited by claims 1, 2, and 9. As Coker in view of Rosenberg, further in view of Teramoto teaches these claims, claim 10 is rejected for similar rationale.

Claims 12-20 present a broader recitation of claims 1-9. These claims are rejected for rationale similar to that given above for claims 1-9 as being obvious over Coker in view of Rosenberg, further in view of Teramoto.

Regarding claim 23, Coker teaches that at least on portion of the boot code is stored internally in said microcontroller [*"In other words, an ECS normally executes software..."* (column 1, lines 29-39); As noted above, Coker teaches no means by which the in-circuit emulation system may access the code on the target-ECS.].

4. Claim 21 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Coker in view of Rosenberg, further in view of Teramoto as applied to claim 12 above, and further in view of "Emulation of the Sparcle Microprocessor with the MIT Virtual Wires Emulation System" by Matthew Dahl, Jonathan Babb, Russel Tessier, Silvina Hanono, David Hoki, and Anant Agarwal (Dahl) and further in view of "A Reconfigurable Logic Machine for Fast Event-Driven Simulation" by Jerry Bauer, Michael Bershteyn, Ian Kaplan, and Paul Vyedin (Bauer).

Coker in view of Rosenberg, further in view of Teramoto does not expressly teach that the shadow system is implemented on a field programmable gate array (FPGA).

Dahl teaches that it is known in the art to emulate a Sparc microprocessor using an FPGA (abstract).

Bauer teaches that hardware emulation can increase simulation speed by up to 10,000 times (introduction, paragraphs 1-2).

Therefore it would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine these teachings and arrive at the decision to implement the shadow system of Coker on an FPGA to realize an enormous increase in simulation speed. Knowledge that this was possible is provided by Dahl, and motivation to combine the references, to increase simulation speed, is provided by Bauer.

Therefore it would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine the teachings of Coker in view of Rosenberg, further in view of Teramoto with Dahl and Bauer to arrive at the invention specified in claim 21.

5. Claim 22 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Coker in view of Rosenberg, further in view of Teramoto as applied to claim 1 above, and further in view of US Patent No. 4,757,534 to Matyas et al.

Regarding claim 22, Coker in view of Rosenberg, further in view of Teramoto does not disclose that the boot code comprises serial numbers and passwords.

Matyas teaches code that comprises serial numbers, passwords, and algorithms [*"In carrying out the computer/smart card protocol, the T output is sent to the smart card together with the parameters P1 (**password**) and P2 (program number | diskette **serial number**, where | denotes concatenation) and a third parameter P3. Where the crypto facility uses the DES **algorithm**, as shown in FIGS. 8 and 9, to encrypt the file key, P3 is the computer number."* (column 13, lines 1-21)].

Additionally, Coker teaches computer software comprising algorithms [*"A shadow system of this invention executes the same software as the target-ECS from system start-up or reset."* (Coker column 2, lines 56-58)]; Rosenberg teaches computer software comprising algorithms [*"Debuggers are quite complex pieces of software. Their inner workings require a suite of sophisticated algorithms and data structures to accomplish their tasks."* (Rosenberg, page 2)]; Teramoto teaches computer software comprising algorithms (Teramoto, FIG. 10).

Matyas and Coker in view of Rosenberg, further in view of Teramoto are analogous art because both are directed to computer software.

It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine the teachings of Matyas with Coker in view of Rosenberg, further in view of Teramoto to include the passwords, serial numbers, and algorithms in the boot code in order to discourage "the copying and sharing of purchased software program" (here, the

boot code) as expressly taught by Matyas (abstract). The combination could be achieved by implementing the smart card cryptographic method taught by Matyas with the target-ECS system taught by Coker.

Therefore, it would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine the teachings of Matyas with Coker in view of Rosenberg, further in view of Teramoto to arrive at the invention specified in claim 22.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jason Proctor whose telephone number is (571) 272-3713. The examiner can normally be reached on 8:30 am-4:30 pm M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Paul Rodriguez can be reached at (571) 272-3753. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jason Proctor
Examiner
Art Unit 2123

jsp

/Zoila E. Cabrera/
Primary Examiner, Art Unit 2123
6/3/08